## CLAIMS

What is claimed is:

1. A method comprising:

reading a line of data from a file containing source code written in a high level language;

generating a stream of tokens from said line of data, said stream of tokens representing any of a specific type of macro in said line of data as being expanded while other types of macros are not expanded;

parsing said stream of tokens;

inserting commands representing operations to be performed by a macro into said stream of tokens if a macro is present; and

writing said stream of token to an output file.

2. The method of claim 1, wherein said generating a stream of tokens further comprises:

determining whether tokens are present in either an input file, a lookahead buffer, or a macro expansion list; and

responsive to finding tokens, reading said tokens first from said lookahead buffer, then from said macro expansion list, then from said input file;

presenting said tokens to a parser so that any macro in said line of data appears to have been expanded.

3. The method of claim 1, wherein said parsing further comprises:

determining a type of token read;

responsive to determining that the token is an end-of-line, processing an input line of tokens;

responsive to determining that the token is not a symbol, adding the token to a current line token list;

responsive to determining that the token is a symbol that indicates a beginning of a macro definition, recording the macro name and macro definition and adding the tokens to a lookahead buffer; and

responsive to determining that the token is a symbol that does not indicate a beginning of a macro definition, adding the token to a current line token list.

4. The method of claim 1, wherein said writing comprises:

writing expanded macro tokens to said output file if said macro is of said specific type of macro; and

writing an original macro call to said output file if said macro is not said specific type of macro.

5. The method of claim 1, wherein said source code written in a high level language comprises a hardware description language (HDL) for representing hardware designs.

6. The method of claim 1, wherein said specific type of macro comprises a scan macro.

7. A method of scan insertion comprising:

reading a hardware description language (HDL) representation of a hardware design, the HDL including a plurality of macro definitions some of which relate to scan insertion;

creating a token stream based on the HDL representation that includes multifaceted tokens that can be hidden from or made visible to a subsequent parsing process by expanding the plurality of macro definitions and making tokens associated with scan macros visible to the subsequent parsing process and marking other tokens as hidden;

performing scan insertion by parsing those of the multifaceted tokens that are visible to the parser and adding appropriate scan commands; and

generating a scan inserted HDL file containing expanded versions of the macro definitions which relate to scan insertion but that omits expanded versions of those that do not relate to scan insertion.

8. The method of claim 7, wherein said HDL comprises a high-level language.

9. The method of claim 7, wherein said hardware design represents an integrated circuit design.

10. A system comprising:

a storage device having stored therein one or more routines for selectively expanding macros within source code; and

a processor coupled to the storage device for executing the one or more routines for selectively expanding macros within source code which, when executing said routine:

reads a line of data from a file containing source code written in a high level language;

generates a stream of tokens from said line of, said stream of tokens representing any of a specific type of macro in said line of data as being expanded while other types of macros are not expanded;

parses said stream of tokens;

inserts commands representing operations to be performed by a macro into said stream of tokens if a macro is present; and

writes said stream of token to an output file.

11. The system of claim 10, wherein said generating a stream of tokens further comprises:

determining whether tokens are present in either an input file, a lookahead buffer, or a macro expansion list; and

responsive to finding tokens, reading said tokens first from said lookahead buffer, then from said macro expansion list, then from said input file;

presenting said tokens to a parser so that any macro in said line of data appears to have been expanded.

12. The system of claim 10, wherein said parsing further comprises:

determining a type of token read;

responsive to determining that the token is an end-of-line, processing an input line of tokens;

responsive to determining that the token is not a symbol, adding the token to a current line token list;

responsive to determining that the token is a symbol that indicates a beginning of a macro definition, recording the macro name and macro definition and adding the tokens to a lookahead buffer; and

responsive to determining that the token is a symbol that does not indicate a beginning of a macro definition, adding the token to a current line token list.

13. The system of claim 10, wherein said writing comprises:

writing expanded macro tokens to said output file if said macro is of said specific type of macro; and

writing an original macro call to said output file if said macro is not said specific type of macro.

14. The system of claim 10, wherein said source code written in a high level language comprises a hardware description language (HDL) for representing hardware designs.

15. The system of claim 10, wherein said specific type of macro comprises a scan macro.

16. A machine-readable medium having stored thereon data representing sequences of instructions, the sequences of instructions which, when executed by a processor, cause the processor to selectively expand macros by:

reading a line of data from a file containing source code written in a high level language;

generating a stream of tokens from said line of , said stream of tokens representing any of a specific type of macro in said line of data as being expanded while other types of macros are not expanded;

parsing said stream of tokens;

inserting commands representing operations to be performed by a macro into said stream of tokens if a macro is present; and

writing said stream of token to an output file.

17. The machine-readable medium of claim 16, wherein said generating a stream of tokens further comprises:

determining whether tokens are present in either an input file, a lookahead buffer, or a macro expansion list; and

responsive to finding tokens, reading said tokens first from said lookahead buffer, then from said macro expansion list, then from said input file;

presenting said tokens to a parser so that any macro in said line of data appears to have been expanded.

18. The machine-readable medium of claim 16, wherein said parsing further comprises:

determining a type of token read;

responsive to determining that the token is an end-of-line, processing an input line of tokens;

responsive to determining that the token is not a symbol, adding the token to a current line token list;

responsive to determining that the token is a symbol that indicates a beginning of a macro definition, recording the macro name and macro definition and adding the tokens to a lookahead buffer; and

responsive to determining that the token is a symbol that does not indicate a beginning of a macro definition, adding the token to a current line token list.

19. The machine-readable medium of claim 16, wherein said writing comprises:

writing expanded macro tokens to said output file if said macro is of said specific type of macro; and

writing an original macro call to said output file if said macro is not said specific type of macro.

20. The machine-readable medium of claim 16, wherein said source code written in a high level language comprises a hardware description language (HDL) for representing hardware designs.

21. The machine-readable medium of claim 16, wherein said specific type of macro comprises a scan macro.

22. A machine-readable medium having stored thereon data representing sequences of instructions, the sequences of instructions which, when executed by a processor, cause the processor to perform scan insertion by:

reading a hardware description language (HDL) representation of a hardware design, the HDL including a plurality of macro definitions some of which relate to scan insertion;

creating a token stream based on the HDL representation that includes multifaceted tokens that can be hidden from or made visible to a subsequent parsing process by expanding the plurality of macro definitions and making tokens associated with scan macros visible to the subsequent parsing process and marking other tokens as hidden;

performing scan insertion by parsing those of the multifaceted tokens that are visible to the parser and adding appropriate scan commands; and

generating a scan inserted HDL file containing expanded versions of the macro definitions which relate to scan insertion but that omits expanded versions of those that do not relate to scan insertion.

23. The machine-readable medium of claim 22, wherein said HDL comprises a high-level language.

24. The machine-readable medium of claim 22, wherein said hardware design represents an integrated circuit design.